PLASMA: Platform for Auxiliary Semantic Modeling Approaches

Alexander Paulus, Andreas Burgdorf, Lars Puleikis, Tristan Langer, André Pomp and Tobias Meisen
Chair of Technologies and Management of Digital Transformation, University of Wuppertal, Wuppertal, Germany

Keywords: Semantic Modeling, Semantic Refinement, Semantic Data Management, PLASMA.

Abstract: In recent years, the impact and usability of semantic data management has increased continuously. Still, one limiting factor is the need to create a semantic mapping in the form of a semantic model between data and a used conceptualization. Creating this mapping manually is a time-consuming process, which especially requires to know the used conceptualization in detail. Thus, the majority of recent approaches in this research field focus on a fully automated semantic modeling approach, but reliable results cannot be achieved in all use cases, i.e., a human must step in. The needed subsequent manual adjustment of automatically created models has already been mentioned in previous works, but has, in our opinion, received too little attention so far. In this paper, we treat the involvement of a human as an explicit phase of the semantic model creation process, called semantic refinement. Semantic refinement comprises the manual improvement of semantic models generated by automated approaches. In order to additionally enable dedicated research in this direction in the future, we also present PLASMA, a platform to utilize existing and future modeling approaches in a consistent and extendable environment. PLASMA aims to support the development of new semantic refinement approaches by providing necessary supplementary functionalities.

1 INTRODUCTION

Due to the emergence of large amounts of data and its usage in machine learning processes, the management of such data has become increasingly important in recent years. Although access to data is often possible, finding and understanding the needed data sources to extract information can be a challenging task without meta information associated to the raw data sets. Semantic data management in the form of Ontology-based Data Management (OBDM) has already laid the foundation for managing heterogeneous data sources more efficiently. Advances in this field of research can be observed in (Linked) Open Data and the introduction of Enterprise Knowledge Graphs(Galkin et al., 2016).

There are two important processes that ensure the quality and acceptance of semantic data management: the creation and maintenance of conceptualizations as well as the description of data sources based on such conceptualizations in form of semantic models. A semantically annotated data source can be found using the conceptual representation of the data and processed using the provided context information stored in the model or the conceptualization. In recent years, research has been focused on reducing the effort that arises when it comes to defining those conceptualizations and the semantic models (cf. Section 2). However, there are no approaches that generally yield flawless results, often requiring a user to supervise the creation and apply modifications. Those modifications can be corrections as well as enhancements of the semantic models. Although mentioned in previous publications such as (Taheriyan et al., 2016), in our opinion, this phase of the modeling process has not yet received sufficient attention.

This phase, which is referred to simply as *refinement* by (Taheriyan et al., 2016), follows the automated semantic modeling. It allows the user to improve an initial version proposed by an automatic modeling approach by manually correcting or enhancing the state of the model. However, a clear definition of this phase has not yet been proposed, which we attempt in this contribution.

Furthermore, to support the development of algorithms for the refinement phase, we also propose PLASMA, a modular platform to integrate approaches for the entire process of creating a semantic model, including the refinement, but also labeling and modeling (cf. Section 2). Using PLASMA, users are enabled to test approaches in a pre-configured environment. For that, PLASMA allows users to ex-

Paulus, A., Burgdorf, A., Puleikis, L., Langer, T., Pomp, A. and Meisen, T.
PLASMA: Platform for Auxiliary Semantic Modeling Approaches.
DOI: 10.5220/0010499604030412
In *Proceedings of the 23rd International Conference on Enterprise Information Systems (ICEIS 2021) - Volume 2*, pages 403-412
ISBN: 978-989-758-509-8

change single components of the semantic model creation process in order to test and evaluate new or even already existing approaches in the same system.

Our contributions in this paper are as follows: First, we introduce the semantic refinement as a phase of the semantic model creation process and state the characteristics of this phase. Second, we present the concept of auxiliary approaches and how those can be utilized to build a comprehensive semantic modeling environment. Third, we release PLASMA to serve as a backbone for future semantic modeling that includes a refinement UI.

In the further course of the paper, we discuss related work in Section 2, then we present our concept of semantic refinement as its own phase in Section 3. Afterwards, we introduce PLASMA in Section 4, followed by current and future usage scenarios of the platform in Section 5. We conclude and give an outlook in Section 6.

2 RELATED WORK

In the past years, several approaches for performing the creation of semantic models have emerged. The process of creating a semantic model has been divided into two phases so far. A basic annotation of a data source can be achieved by performing a semantic labeling step, followed by the semantic modeling step, which aims to relate initially mapped concepts and equip the annotation with additional context (cf. (Vu et al., 2019)).

Semantic labeling, which is the initial mapping from labels (e.g., table headers or leafs in a structured data set) to a conceptualization (e.g., ontology), is mainly performed by following either a label-driven or a data-driven strategy. Examples for label-driven mappings have been proposed by (Polfliet and Ichise, 2010), (Pinkel et al., 2017), (Paulus et al., 2018) or (Papapanagiotou et al., 2006). Next to those, there also exist data-driven mapping approaches that rely mainly on using the data values of a data set. For instance, (Syed et al., 2010) as well as (Wang et al., 2012) proposed approaches that provide labels using the data values within a data set in conjunction with external (knowledge) databases to identify the most reasonable concept. (Ramnandan et al., 2015),(Abdelmageed and Schindler, 2020) use algorithmic approaches whereas (Pham et al., 2016), (Rümmele et al., 2018), (Chen et al., 2019), (Hulsebos et al., 2019) as well as (Takeoka et al., 2019) and (Pomp et al., 2020) use machine learning approaches to solve the task.

All previously mentioned semantic labeling approaches rely on a predefined conceptualization, although approaches like (Pham et al., 2016), (Pomp et al., 2020) or (Jiménez-Ruiz et al., 2015) are able to create concepts during the mapping process by integrating the user into the labeling process. Using such approaches in platforms, like Optique (Giese et al., 2015), it becomes possible to redefine the found mappings using an R2RML editor (Sengupta et al., 2013).

Following the semantic labeling phase, the initial mappings can be extended by additional meaningful concepts as well as selecting the most suitable relationships that hold between the different concepts, resulting in a semantic model for that data set. Approaches presented by (Knoblock et al., 2012) and (Taheriyan et al., 2013; **?**; **?**) as well as (Uña et al., 2018) have shown significant improvements in this area during the last years. Most recent approaches like (Vu et al., 2019) and (Futia et al., 2020) show the ongoing research interest in the topic. However, none of the approaches does solve the task perfectly, due to ambiguities in the data sets or the encounter of unseen cases in the training data.

To finalize the model and correct potential errors, human interaction is needed. However there are only few approaches that feature tools to inspect and modify a semantic model after the automated generation. A first approach to support semantic model modification was Karma, presented by (Knoblock et al., 2012) and (Gupta et al., 2012). Karma is described as a data integration tool that enables users to feed in data and create a semantic model for it. The used automated semantic modeling approach is based on the method developed by Taheriyan et al. Karma has been used in various projects (cf. (Szekely et al., 2015)). A similar approach to Karma is followed by ESKAPE(Pomp et al., 2017), a semantic data platform for enterprises, handling the full data management process from ingestion to extraction. However, ESKAPE only performs basic semantic labeling automatically - the semantic modeling has to be done manually by the users. Furthermore, there exist multiple commercial tools like PoolParty[1] or Grafo[2] that focus on creating knowledge graphs but are not available for free use and furthermore are not extendable to support own algorithms.

Both Karma and ESKAPE are capable of doing a data schema analysis, analyzing live input data to extract labels and structure to perform the labeling step on. Both also support multiple data formats during this phase as well as provide a GUI (cf. (Szekely et al., 2015) and (Pomp et al., 2018)), which allows users to modify the semantic model. However, al-

[1] http://poolparty.biz
[2] http://gra.fo

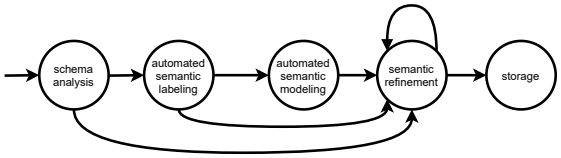

Figure 1: Overview of the phases during the creation of a semantic model with the *semantic refinement* phase following the *(automated) semantic modeling*.

though following the human in the loop approach, both systems do not actively support the user during the modification phase.

From the observed state of the art, the main process of creating a semantic model is depicted in Figure 1. We identify two main phases, *automated semantic labeling* and *automated semantic modeling* that have received significant research during the past years, and the subsequent refinement phase featuring the interactive modeling. Also, as those stages are provided by modeling systems like Karma, we add a *schema analysis phase* that builds the initial syntactic model from a data set and a final *storage phase* that persists the complete semantic model, e.g., by integrating it into a knowledge graph or semantic storage.

3 SEMANTIC REFINEMENT

While there have been major advances in the field of semantic labeling and modeling, the subsequent refinement has been mentioned but not actively supported. However, systems like Karma and ESKAPE provide the ability to improve the quality of generated models as a fourth phase following the automated semantic modeling. We refer to this phase as *Semantic Refinement*, an interactive and iterative process of improving an initial version proposed by an automatic modeling approach. The terminology is not new, (Paulheim, 2016) used the "refinement" term to describe the automated improvement of knowledge graphs, whereas (Futia et al., 2020) use it to describe the final automated validation of their semantic model computation. However, in both cases this is an automated procedure and which, in the first case is disconnected from the semantic model creation process and in the latter case we associate it as a part of the semantic modeling phase. Also, the refinement of a semantic model has been mentioned in previous publications (e.g., (Szekely et al., 2013), (Taheriyan et al., 2015) and (Taheriyan et al., 2016)), but was often not in the focus and being referred to as a necessity than a dedicated phase of the semantic model creation process. However, it is unclear if the authors referred to an automated step like proposed by (Futia et al., 2020) or a fully user-based adjustment as (Pomp et al., 2017) propose.

We see the main characteristic of the dedicated semantic refinement phase as the manual use of a tool to improve the quality of a semantic model. The user is kept involved by inspecting the result after each modification. Possible user operations in this phase are adding new concepts or relations from the conceptualization, deleting or replacing existing elements or modifying single elements if permitted by the system (e.g., changing properties). Even extending the conceptualization (c.f. (Pomp et al., 2019)) is a valid operation if the system used follows the open world assumption and lets the user add own facts to the models which are not yet present in ontologies.

In our opinion, an essential part of the semantic refinement phase is a continuous support by the system, aiding the user on how to improve the model by taking the manual load of checking, validating, correcting and extending the contents, e.g., by solving discrepancies identified during the modeling phase. Therefore, we propose that algorithms are used repeatedly after each user-created modification. Such algorithms use the current state of the semantic model as input to evaluate and recommend one or multiple possible changes to the semantic model. For the output, we propose that approaches yield suggestions on how to incrementally improve the semantic model by providing single relations or concepts as well as subgraphs to be added. For each iteration, the user can then also decide to accept or reject those changes, possibly updating the model and triggering a new recommendation iteration. Of course, the user can also ignore the recommendations and apply own further changes.

Examples for refinements span selection or exchange of ambiguous or mutually exclusive concepts, e.g.. for specifying units of measure, as well as adding new concepts that were either not included in the automatically generated model for various reasons, most commonly due to being unseen in the past. Also, refinements could include adding new and more appropriate concepts that were unavailable to previous automated steps, e.g., by using external data sources. The result should resemble the way the user interprets the data set as close as possible.

4 PLATFORM FOR AUXILIARY SEMANTIC MODELING APPROACHES

With the new process model in mind, a platform is needed which supports and focuses manual semantic

refinement alongside all other phases. We designed the *PLatform for Auxiliary Semantic Modeling Approaches* (PLASMA) as a platform for scientific research that supports the whole semantic model creation process and aim to make it publicly available for scientific purposes [3].

With the release of PLASMA as a new platform, we aim to build a baseline system for semantic model creation, while reducing the effort for the initial setup and maintenance to a minimum. The main driving factor is the ability to support all five phases of the semantic model creation process (Figure 1) in one platform. This especially includes the semantic refinement phase introduced in Section 3. We also chose a modular approach to be able to exchange single components and connect new ones using a defined interface, while changing as little of the surrounding system as possible. This would also apply when connecting existing systems like Serene (Uña et al., 2018) or SeMi (Futia et al., 2020).

This approach is inspired by (Uña et al., 2018), which connected Karma to their system to achieve comparability, as well as the Serene Benchmark (Rümmele et al., 2018), which focuses on integrating different fully automated semantic labeling approaches. PLASMA also allows the utilization of those approaches but encapsules them in a modular way making them exchangeable. Furthermore, instead of solely focusing on the semantic labeling phase, PLASMA provides a complete modeling environment, including the semantic modeling and refinement phase. PLASMA also provides common functionalities like schema analysis and a graphical modeling UI to not require each developer to implement those each time they design a new approach. Although being provided by the system, those components are also to be realized as modules, making them exchangeable if required.

4.1 Components

In the following, we present the single components of the architecture separately and outline dependencies and requirements. Also, a detailed feature overview is given for each service. Figure 2 gives a schematic overview of the components and their interactions and dependencies.

4.1.1 Data Model

The data central model of PLASMA is called a *Combined Model*. It holds the syntactic model as well as the current state of semantic model that is created. We define the *syntactic model* as $M^{Syn} = (nodes, edges, root)$ where *nodes* are general nodes, representing an object or a collection, or leaf nodes representing data labels and their values. The root node is stored in *root*. The set *edges* contains all the edges that connect the nodes to represent the structure of the data source so that especially nested structures are preserved. Given a semantic conceptualization, like an ontology $O = (C,R)$, where C denotes available concepts and R the relations between them, a semantic model is defined as $M^{Sem} = (C',R')\ C' \subseteq C, R' \subseteq R$. The combined model $M^C = (M^{Syn}, M^{Sem}, L)$ then denotes a fusion of the syntactic and semantic model. It contains an additional mapping $L : C' \rightarrow nodes$ mapping single concepts $C' \in M^{Sem}$ from the semantic model to attributes in the syntax model M^{Syn}. As L is a injective mapping, it must hold that $\nexists\ c_1, c_2 \in C'$ where $L(c_1) = L(c_2)$ (only one concept is assigned to each syntax node).

Furthermore, the combined model defines a format for *modifications* to the model. A modification for a combined model $M^C = (M^{Syn}, M^{Sem}, L)$ is defined as $\Delta = (C^R, A, R^R, \omega)$, where $C^R \subseteq C$ and $R^R \subseteq R$ denote the sets of modified (added or adjusted or deleted) concepts and and relations, respectively. The set A denotes the anchor points, that is $A = \{a \mid a \in C^R \wedge (a \in C',\ where\ C' \in M^{Sem} \vee \exists\ n \in nodes, L(a) = n)\}$. This requires the contained subgraph to be connected to the existing M^{Sem} or to contain at least one concept $a \in C^R$ being mapped to an attribute. Those modifications can either be applied directly to M^C or stored for the user to chose which ones to apply. In this case, we refer to them as *Recommendations* and require $A \neq \emptyset$, not allowing disconnected subgraphs and indicate an (optional) confidence score with $\omega \in [0,1]$, indicating on how certain the issuer of the modifications is that this will indeed improve the state of the model.

4.1.2 Data Source Service (DSS)

Before the user can start modeling, a *data source* has to be created using the DSS. A data source is the basic reference unit of PLASMA. We define a data source as a tuple $DS = (name, note, desc, data)$, where *name* is the unique name of that source, *note* is a short description of the data source, *desc* is a textual description of the data and *data* is the later provided sample input data, being empty on creation. The *desc* is an extended textual description of the contained data that can be provided to supply meta information for labeling and modeling approaches.

[3] https://github.com/tmdt-buw/plasma

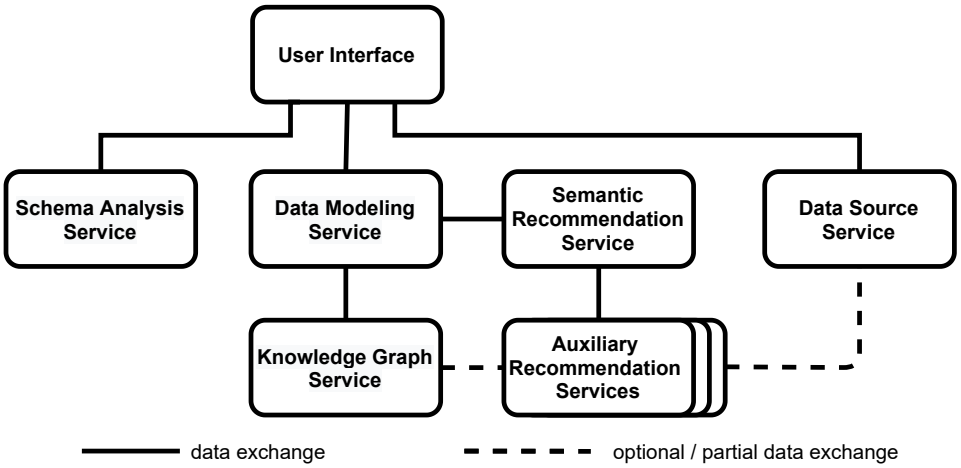

Figure 2: Architectural overview of the PLASMA's components. All connections indicate communicating components.

4.1.3 Schema Analysis Service (SAS)

Once a data source has been created, we begin the semantic model creation process with phase 1, the schema analysis. Sample data points need to be inserted for a data source to analyze them and extract the structure and labels. Thus, the SAS takes a data source DS_x and a set of sample data points $SDP^x = P_1^x, P_2^x, ..., P_n^x$ to analyze the data. After the analysis, the SAS identifies the data structure and provides a *syntactic model* M_x^{Syn}.

4.1.4 Auxiliary Recommendation Services (ARS)

The Auxiliary Recommendation Services (ARS) are services that provide modeling recommendations during different stages of the semantic model creation process. Each ARS encapsulates an algorithm to perform a specific task in either the labeling (phase 2), modeling (phase 3) or refinement phase (phase 4), being denoted *ARS for Labeling (ARS-L)*, *ARS for Modeling (ARS-M)* and *ARS for Refinement (ARS-R)*, respectively, based on their type. An ARS receives the current state of the combined model and optionally some meta information and will return a combined model which has been modified or supplemented with some recommendations $\Delta_1, .., \Delta_n$ for the user to chose from. The services are using a standardized interface to be quickly exchangeable and allow PLASMA to support multiple approaches on how to solve the task for each phase.

The ARS are modular services, thus posing as little limitations as possible to developers on what technology stack to use when implementing their solutions. As long as an ARS satisfies the given interface provided by the platform, it can be connected. Using a decoupled architecture, new approaches can be developed and tested on their own before being attached to PLASMA. If the approach to be connected by the ARS itself is an already implemented and running system with a predefined API, the ARS can also function as a proxy to it, e.g., by using an offered interface and converting communication to match PLASMA's data format (see Section 4.1.1). A single service can be registered to perform operations for different phases, e.g., by providing multiple interfaces. The three types of ARS are:

- ARS for Labeling (ARS-L): receive a data source DS_x and the syntactic model M_x^{Syn} and provide an initial combined model by performing the semantic labeling, returning a combined model M_x^C with labeled leaf nodes.
- ARS for Modeling (ARS-M): receive DS_x and M_x^C and perform the semantic labeling. Return M^C including the initial semantic model or multiple possible models represented as recommendations.
- ARS for Refinement (ARS-R): receives DS_x and M_x^C and computes modifications that might help improve the state of the model (cf. Section 3). How this is achieved and which modifications are deemed fitting is up to the contained algorithm. Return M^C and a number of optional modifications $\Delta_1, .., \Delta_n$ as recommendations. ARS-R do not alter M_x^{Syn} or M_x^{Sem} directly. Also, ARS-R are expected to be invoked multiple times and thus may keep an internal state for the specific data source, e.g., to improve recommendations in follow-up executions.

4.1.5 Semantic Recommendation Service (SRS)

This service coordinates the ARS that are connected to PLASMA to perform their tasks during phases 2-4 of the semantic model creation process. Therefore, the SRS provides three different endpoints to request modifications for phase 2-4 of the semantic model creation process. All ARS of one type use the same interface. Using standardized interfaces allow the SRS to contact previously unknown services via auto discovery, a feature that will be operational in the near future. In the current state, the SRS is configured which of the available ARS to use. Once development of a new ARS has finished, the ARS can be integrated into the architecture and the SRS adjusted to use the new ARS for further operations.

4.1.6 Data Modeling Service (DMS)

This service keeps the current state of all combined models M^C in its database and updates them once modifications are issued. Whenever an actor modifies the current state of the combined model, those changes are processed and validated using the DMS. These modifications, which are done by applying a modification Δ to a combined model M^C, can either be made by the user using the UI or originate from (automated) recommendation systems, like ARS. The only restriction is that there always has to be one defined state of combined model M^C_x for each data source DS_x in the DMS to be provided to the UI.

4.1.7 Knowledge Graph Service (KGS)

This service is the platform's main source for semantic information. It maintains the conceptualization that is used for generating mappings as well as the mappings itself in the form of semantic models extracted from the combined model. We refer to those types as conceptualization layer and mapping layer.

To represent these two layers, we use an upper ontology as introduced by Pomp (Pomp, 2020). This ontology allows to combine the conceptualization and the mapping layer by representing them in a single knowledge graph. By utilizing a knowledge graph, we achieve direct linkage between the conceptualization and the mapping layer, allowing algorithms to efficiently operate on the structure.

The conceptualization layer can be initialized by importing already existing ontologies in RDF Turtle syntax to form the target conceptualization for the semantic models. During the import, RDF triples are converted into Concepts and Relations as defined by the upper ontology while keeping provenance should those later be exported again. Both the conceptualization as well as semantic models can be exported as RDF in Turtle syntax again. Semantic models can be created, deleted and searched.

4.2 User Interface

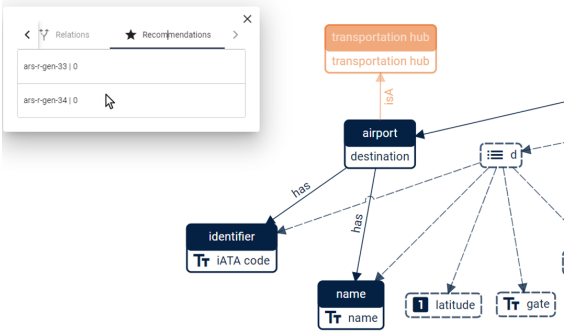

Figure 3: Partial screenshot of the refinement UI of PLASMA during the modeling process with an unfinished semantic model. Highlighted node is a recommendation.

To allow the user to inspect the processing that has taken place, we provide a web-based user interface (UI) that allows to operate the basic functionalities of PLASMA. In the following, we give a brief overview of the UI, including the available views and functionalities. PLASMA's UI consists of multiple views, one for the creation of data sources, one for the submission of sample data for an existing data source and a refinement UI that enables the user to perform manual actions during the semantic refinement phase. The realized modeling concept for the refinement UI follows the approach presented by (Pomp et al., 2018), but has received a major overhaul. Figure 3 shows the refinement UI displaying the syntactic model and an unfinished semantic model. Here, the user can inspect recommendations issued by the system and edit the current state of the semantic model. The refinement UI aims to provide the user with a convenient modeling environment, hiding technical details for beginner users. The syntactic model is shown as nodes with dashed lines and can be modified by the user, i.e., nodes can be rearranged. For hierarchical data, the UI also displays the structure. Based on the environment the refinement UI is used (cf. Section 5), even more sophisticated operations, like syntactic node splitting or assigning data types, can be realized. However, as this requires modifications to the original data, further data processing components are needed that are not part of PLASMA itself.

The semantic model M^{Sem} consists of nodes that are comprised of two parts. A *concept part* (upper half), realizing an assigned concept from the conceptualization layer and a *mapping part* (lower half)

representing the mapping layer component (cf. Section 4.1.7). Each mapping part serves as a projection of a concept (denoted in the concept part) in a semantic model. While concept parts cannot be edited or adjusted, the mapping part is customizable, e.g., can be renamed in order to resolve ambiguities within the model, while still being bound to a concept which ensures universal understandability. Relations are taken directly from the conceptualization layer and can therefore not be modified. Available elements can be found using the left side drawer offering a search functionality for existing concepts and relations. Concepts as well as relations are then added using drag and drop mechanisms. When a concept is attached to a leaf node of the syntactic model, a mapping is created between the semantic model and the data set in L_x.

Following each modification, the DMS requests an update on the recommendations from the SRS using the now modified model as input. This may result in recommendations being removed as they no longer apply, or new ones being generated. Recommendations are displayed when hovering one in the recommendations tab and applied by double-clicking on them, providing a convenient way to extend or modify the semantic model. However, as we are experimenting with different visualization techniques, this might change in the future. Figure 3 shows recommendations as highlighted nodes, in this instance a possible generalization of an *airport* as a *transportation hub* to indicate it as an organizational unit instead of, for example, the area. There might be more recommendations available, but are hidden due to a general threshold for certainty of $\omega \geq 0.5$, resulting in recommendations with lower certainty to be hidden.

4.3 Semantic Model Creation Process in PLASMA

In this section, we briefly describe the main phases of the semantic model creation process (Figure 1) and how those are executed in PLASMA. Figure 4 gives an overview of the main operations and their order. For an existing data source *DS*, data can be added to be analyzed. The system currently processes JSON typed example data values as JSON can be used to describe both structured and flat data sources. The data is then transmitted to the SAS where it is analyzed ①. Once the syntactic model M^{Syn} has been created (phase 1), the result is transmitted to the DMS where an empty semantic model M^{Sem} is attached to it ②. The model is then handed to the SRS for the initial labeling and modeling steps ③. To achieve this, the SRS requests an automated semantic labeling (phase 2) from an ARS-L. The result is an initial labeling of the leaf nodes of the syntactic model in the returned combined model M^C ④. This is followed by requesting an automated semantic modeling (phase 3) to extend M^C past the labeled concepts. The ARS-M performs this operation and returns the M^C to the SRS ⑤. This model is then returned to the DMS ⑥ and shown in the User Interface ⑦. Once the user performs a manual action, the updated M^C is handed over to the DMS ⑧ which then requests a recommendation for the current M^C from the SRS ③, using the refinement endpoint. The SRS uses instances of ARS-R to acquire one or more recommendations ⑨ and returns those to the DMS ⑩ where M^C is adjusted and the UI updated ⑦, giving the user the ability to inspect and optionally accept those recommendations. Steps ⑧ ③ ⑨ ⑩ ⑦ are repeatable and form the *Semantic Refinement Process* (phase 4). Once the user has finished editing, the finalized semantic model M^{Sem} is sent to the KGS for storage (phase 5), ending the refinement process and bringing the semantic model creation of the given data source to an end ⑪.

5 ONGOING USE CASE

PLASMA was designed to be a baseline support tool for upcoming projects that aim to improve the semantic model creation process. Future approaches can be developed and tested in a controlled environment, where the auxiliary services architecture supports a plug and play approach to setup and test new approaches after PLASMA is set up. PLASMA is currently in use in the Bergisch.Smart.Mobility[4] research project, which features a data marketplace for smart city open data, providing the semantic model creation functionalities for the semantic annotation of data sources. Within the ongoing project, PLASMA is integrated alongside the semantic data ingestion engine and provides an infrastructure that enables municipal employees to create semantic models to describe their data sets. The created semantic models are then used to integrate the data and provide it to data consumers. Figure 5 shows how PLASMA is positioned in the general workflow of the marketplace. During the project runtime, PLASMA's components are adjusted to improve the usability of semantic technologies for municipal workers. There are also multiple ARS being tested to improve the overall automatic modeling performance of the platform, for example one focusing on frequently used geospatial concepts.

[4]https://www.bergischsmartmobility.de/en/the-project/

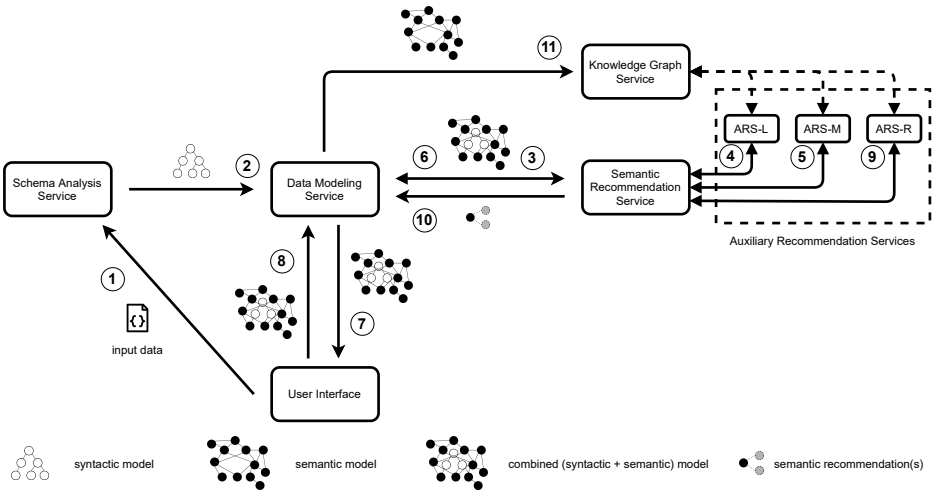

Figure 4: The semantic model creation process. ① Upload of input data to the SAS. ② Submission of the resulting syntactical model to the DMS. ③ Request for semantic recommendations based on the current state of the combined model. ④ Requesting of semantic labeling using separated *Auxiliary Recommendation Service* (ARS). ⑤ Requesting of semantic modeling using separated ARS. ⑥ Provision of initial model to the DMS. ⑦ Transmission of current model to GUI. ⑧ User manually updates model. ⑨ Requesting of semantic recommendation using separated ARS. ⑩ Provision of one or more suggestions to the DMS. ⑪ Submission of final semantic model to the KGS for storage or export.

5.1 Future Usage Scenarios

In the project, PLASMA is evolved to support the users when creating semantic models. However, aside from those advancements, estimating the quality of different approaches is also in the focus of the platform. For example, to compare two different semantic modeling approaches, after providing the target ontology and connecting any ARS-L for the labeling phase, approach *A*, realized as an ARS-M, would be connected to the SRS. The model is created and afterwards the service exchanged to another ARS-M containing approach *B*. After another run, two results are available for export and evaluation. By exchanging only one component during runs, comparable results can be reached as the remaining environment remains unchanged.

With the UI including the recommendation inspection and selection, PLASMA also aims to provide a platform to advance research towards user support during the semantic refinement phase. A specific aim of PLASMA is to enhance the visualization and selection of recommendations to actively support the refinement process. Therefore, new approaches can be developed utilizing PLASMA as a platform.

6 CONCLUSION AND OUTLOOK

In this paper, we introduced the interactive manual refinement of a semantic model as an explicit phase of the semantic model creation process. It covers an essential part of the creation of semantic models using a human in the loop approach to manually correct and improve results generated by automated approaches. We also presented PLASMA, an extendable platform aimed to support the development and integration of semantic labeling/modeling/refinement approaches using the concept of auxiliary modeling services. The platform is designed to function as a tool to develop new approaches for the semantic refinement process. PLASMA is build in a modular pattern to (i) allow modifications or replacements of single components, to (ii) integrate new approaches and evaluate them using a common framework and to (iii) inspect results using a web based UI aimed to support future semantic model refinement approaches.

In the future, we plan to provide a set of existing modeling algorithms for PLASMA by implementing current approaches in dedicated ARS. The SRS is planned to support multiple similar approaches, e.g., for semantic labeling, and let the user decide which algorithm to use for the current run. This would greatly reduce the manual configuration currently needed and also allow users to evaluate and compare different approaches without the need to modify the technical architecture during runs. In addition, we plan to support batch operations, allowing to analyze or model multiple data sources automatically, optionally using different ARS instances for each run. This way, we hope to eventually utilize PLASMA as an evaluation framework for different

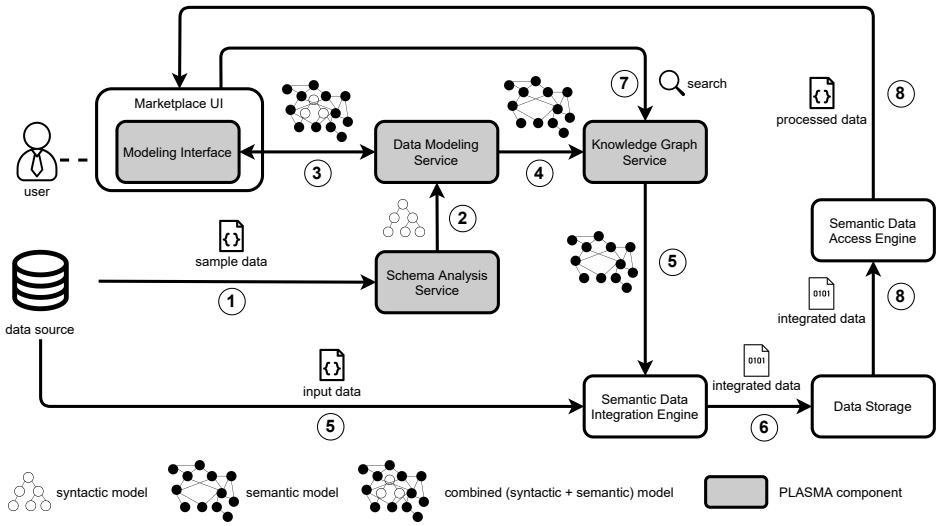

Figure 5: Ingestion and retrival process for a single data source with PLASMA integrated into a data marketplace. The SRS is omitted for brevity. ① Upload of sample data to the SAS. ② Submission of the resulting syntactical model to the DMS. ③ Creation / refinement of the semantic model for the data source. ④ Submission of final semantic model to the KGS. ⑤ Integration of the raw input data using the created semantic model. ⑥ Storage of semantically annotated integrated data in the data storage. ⑦ User(s) search data space based on semantic models. ⑧ Raw data or integrated data is requested, processed and transferred to the user.

automated modeling approaches. In this context, to even save the need to export and evaluate models, a integrated benchmarking functionality is envisioned, but needs further specification.

REFERENCES

Abdelmageed, N. and Schindler, S. (2020). JenTab: Matching Tabular Data to Knowledge Graphs. The 19th International Semantic Web Conference.

Chen, J., Jimenez-Ruiz, E., et al. (8/10/2019 - 8/16/2019). Learning Semantic Annotations for Tabular Data. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence*.

Futia, G., Vetrò, A., and de Martin, J. C. (2020). SeMi: A SEmantic Modeling machIne to build Knowledge Graphs with graph neural networks. *SoftwareX*, 12:100516.

Galkin, M., Auer, S., and Scerri, S. (2016). Enterprise Knowledge Graphs: A Backbone of Linked Enterprise Data. In *2016 IEEE/WIC/ACM International Conference on Web Intelligence (WI)*.

Giese, M., Soylu, A., Vega-Gorgojo, G., Waaler, A., Haase, P., Jimenez-Ruiz, E., Lanti, D., Rezk, M., Xiao, G., Ozcep, O., and Rosati, R. (2015). Optique: Zooming in on Big Data.

Gupta, S., Szekely, P., et al. (2012). Karma: A System for Mapping Structured Sources into the Semantic Web. In *Extended Semantic Web Conference*.

Hulsebos, M., Hu, K., et al. (2019). Sherlock. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*.

Jiménez-Ruiz, E., Kharlamov, E., et al. (2015). BootOX: Bootstrapping OWL 2 Ontologies and R2RML Mappings from Relational Databases. In *International Semantic Web Conference (Posters & Demos)*.

Knoblock, C. A., Szekely, P., et al. (2012). Semi-Automatically Mapping Structured Sources Into the Semantic Web. In *Extended Semantic Web Conference*, pages 375–390.

Papapanagiotou, P., Katsiouli, P., et al. (2006). RONTO: Relational to Ontology Schema Matching. *AIS Sigsemis Bulletin*, 3(3-4):32–36.

Paulheim, H. (2016). Knowledge graph refinement: A survey of approaches and evaluation methods. *Semantic Web*, 8(3):489–508.

Paulus, A., Pomp, A., et al. (2018). Gathering and Combining Semantic Concepts from Multiple Knowledge Bases. In *ICEIS 2018*, pages 69–80, Setúbal, Portugal.

Pham, M., Alse, S., et al. (2016). Semantic Labeling: A Domain-Independent Approach. In *The Semantic Web – ISWC 2016*, pages 446–462, Cham. Springer International Publishing.

Pinkel, C., Binnig, C., et al. (2017). IncMap: a Journey Towards Ontology-based Data Integration. *Datenbanksysteme für Business, Technologie und Web (BTW 2017)*.

Polfliet, S. and Ichise, R. (2010). Automated Mapping Generation for Converting Databases into Linked Data. In *Proceedings of the 2010 International Conference on Posters & Demonstrations Track*.

Pomp, A. (2020). *Bottom-up Knowledge Graph-based Data Management*. Berichte aus dem Maschinenbau. Shaker.

Pomp, A., Kraus, V., et al. (2020). Semantic Concept Recommendation for Continuously Evolving Knowledge

Graphs. In *Enterprise Information Systems*, Lecture Notes in Business Information Processing. Springer.

Pomp, A., Lipp, J., and Meisen, T. (2019). You are Missing a Concept! Enhancing Ontology-based Data Access with Evolving Ontologies. In *Proceedings, 13th IEEE International Conference on Semantic Computing*, pages 98–105.

Pomp, A., Paulus, A., et al. (2018). A Web-based UI to Enable Semantic Modeling for Everyone. *Procedia Computer Science*, 137:249–254.

Pomp, A., Paulus, A., Jeschke, S., and Meisen, T. (2017). ESKAPE: Platform for Enabling Semantics in the Continuously Evolving Internet of Things.

Ramnandan, S. K., Mittal, A., et al. (2015). Assigning Semantic Labels to Data Sources. In *The Semantic Web. Latest Advances and New Domains*, pages 403–417, Cham. Springer International Publishing.

Rümmele, N., Tyshetskiy, Y., and Collins, A. (2018). Evaluating Approaches for Supervised Semantic Labeling. *CoRR*, abs/1801.09788.

Sengupta, K., Haase, P., Schmidt, M., and Hitzler, P. (2013). Editing r2rml mappings made easy. In *Proceedings of the 12th International Semantic Web Conference*, page 101–104.

Syed, Z., Finin, T., et al. (2010). Exploiting a Web of Semantic Data for Interpreting Tables. In *Proceedings of the Second Web Science Conference*.

Szekely, P., Knoblock, C. A., et al. (2013). Connecting the Smithsonian American Art Museum to the Linked Data Cloud. In *The Semantic Web: Semantics and Big Data*, pages 593–607, Berlin, Heidelberg. Springer Berlin Heidelberg.

Szekely, P., Knoblock, C. A., et al. (2015). Building and Using a Knowledge Graph to Combat Human Trafficking. In *The Semantic Web - ISWC 2015*, volume 9367, pages 205–221. Springer, Cham.

Taheriyan, M., Knoblock, C. A., et al. (2013). A Graph-Based Approach to Learn Semantic Descriptions of Data Sources. In *The Semantic Web – ISWC 2013*, pages 607–623, Berlin, Heidelberg. Springer Berlin Heidelberg.

Taheriyan, M., Knoblock, C. A., et al. (2015). Leveraging Linked Data to Infer Semantic Relations within Structured Sources. In *Proceedings of the 6th International Workshop on Consuming Linked Data (COLD 2015)*.

Taheriyan, M., Knoblock, C. A., et al. (2016). Learning the Semantics of Structured Data Sources. *Journal of Web Semantics*, 37-38:152–169.

Takeoka, K., Oyamada, M., et al. (2019). Meimei: An Efficient Probabilistic Approach for Semantically Annotating Tables. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01):281–288.

Uña, D. D., Rümmele, N., et al. (2018). Machine Learning and Constraint Programming for Relational-To-Ontology Schema Mapping. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence*.

Vu, B., Knoblock, C., and Pujara, J. (2019). Learning Semantic Models of Data Sources Using Probabilistic Graphical Models. In *The World Wide Web Conference*, WWW '19, pages 1944–1953, New York, NY, USA. ACM.

Wang, J., Wang, H., et al. (2012). Understanding Tables on the Web. In *Conceptual Modeling*, pages 141–155, Berlin, Heidelberg. Springer Berlin Heidelberg.